UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/082,794	02/22/2002	David Bau III	41016.P008	2046

25943 7590 04/30/2007

SCHWABE, WILLIAMSON & WYATT, P.C.
PACWEST CENTER, SUITE 1900
1211 SW FIFTH AVENUE
PORTLAND, OR 97204

EXAMINER	
RUTTEN, JAMES D	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
04/30/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/082,794	BAU ET AL.	
	Examiner	Art Unit	
	J. Derek Rutten	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *19 April 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-10;12-18,20-22,31-40 and 42-44* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-10,12-18,20-22,31-40 and 42-44* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

1. A request for continued examination under 37 CFR 1.114 was filed in this application after a decision by the Board of Patent Appeals and Interferences, but before the filing of a Notice of Appeal to the Court of Appeals for the Federal Circuit or the commencement of a civil action. Since this application is eligible for continued examination under 37 CFR 1.114 and the fee set forth in 37 CFR 1.17(e) has been timely paid, the appeal has been withdrawn pursuant to 37 CFR 1.114 and prosecution in this application has been reopened pursuant to 37 CFR 1.114. Applicant's submission filed on 4/19/07 has been entered. Claims 1, 10, 12, 16, 31, 32, and 38 have been amended, and claims 11, 19, 23-30, 41, and 45-52 have been canceled. Claims 1-10, 12-18, 20-22, 31-40, and 42-44 remain pending in the application and have been fully considered by the examiner.

Response to Arguments/Amendments

2. Applicant's arguments filed 4/19/07 have been fully considered but they are not persuasive for the reasons set forth below.

3. On page 11, Applicant does not argue in regard to the Double Patenting rejections, but agrees to submit the necessary Terminal Disclaimers upon issuance of 10/082,807, 10/784,492, or the instant application. As such, the rejection is maintained.

4. At the bottom of page 12, filed 4/19/07, Applicants argue:

> In contrast, WebLogic fails to teach or suggest **an integrated development environment** that, in response to user input, automatically specifies declarative annotations that, when recognized by a compiler, cause the compiler to generate one or more persistent components to maintain

> conversational state related to an identified method. Rather, **WebLogic simply teaches a framework for the development and deployment of EJBeans.** In WebLogic, **a user/developer may either obtain or program EJBeans,** which may include business logic to perform a number of methods. [emphasis added]

In the above passage, Applicants acknowledge that WebLogic discloses a "framework for the development and deployment of EJBeans." Such a "framework for development" appears to coincide with the description of an IDE as presented in Applicants' originally filed specification (e.g. see page 10 lines 9-10). Therefore, Applicants' argument appears to acknowledge that WebLogic at least provides an IDE for a "user/developer."

Further arguments on page 13 focus on the "declarative annotations" of the instant invention. On page 13, Applicants essentially argue that the WebLogic reference does not disclose the automatic specification of declarative annotations used by a compiler to generate persistent components. However, WebLogic appears to disclose the automatic specification of declarative annotations as being provided by the DDCreator utility application that "takes a text file specification and creates the appropriate serialized deployment descriptor" (see page 5, 2nd paragraph under "Step 2"). WebLogic further describes that declarative annotations are used by a compiler to generate components (see page 6, e.g. "Step 3. Generate the EJB container classes using ejbc").

Applicants further argue (see bottom of page 13) essentially that the components generated by WebLogic are not "persistent" components that maintain conversational state. Applicants appear to support this by suggesting that WebLogic's EJBeans, while being persistent components, cannot read on the claims since they are not generated in response to WebLogic's deployment descriptors. However, as suggested above and evidenced by WebLogic (i.e. page 6

"Step 3"), the EJBeans are created using the deployment descriptors. Thus, Applicants arguments are not persuasive.

5. On page 15, filed 4/19/07, Applicants essentially argue that Pagé's queues are not instantiated based on any declarative annotations. In response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986). Pagé is not relied upon to show the generation of code based upon declarative annotations. Rather, WebLogic teaches the generation of code based upon declarative annotations, and it would have been obvious to combine the teaching in order to implement "store and forward" technology in order to provide reliable data delivery as suggested by Pagé (see column 2 lines 30-34). Thus, Applicants' arguments are not persuasive.

6. Further arguments on pages 15-19, filed 4/19/07, are based on arguments as addressed above, and are not persuasive for the same reasons.

Double Patenting

7. The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. See *In re Goodman*, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); *In re Longi*, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); *In re Van Ornum*, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970);and, *In re Thorington*, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

A timely filed terminal disclaimer in compliance with 37 CFR 1.321(c) may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent is shown to be commonly owned with this application. See 37 CFR 1.130(b).

Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

8. The text of the provisional rejections of claims 1-52 under the judicially created doctrine of obviousness-type double patenting, as found in the previous Office Action, is reproduced below for convenience.

9. Claim 1-10, 12-18, 20-22, 31-40, and 42-44 are provisionally rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claim 1, 4-10, 12-22, 25-30, 33, 35-38, 4144, 46-52, 54-63, 67-75, 77-80, 83, and 84 of copending Application No. 10/082,807 (hereinafter "the '807 application"). Although the conflicting claims are not identical, they are not patentably distinct from each other because with respect to claim 1, for example, the 2/7/07 filing of the '807 application discloses:

1. A method of specifying a stateful web service within a procedural programming environment, the method comprising:

first facilitating, by an integrated development environment of a computing device, a user in providing a source code representation of at least a portion of web service logic, the logic including one or more methods; See page 2, first clause of claim 1:

> providing a source code representation of at least a portion of web service logic, the logic including at least one method

Also see the preamble of claim 1: "procedural programming environment."

second facilitating, by the integrated development environment of the computing device, the user in identifying one of said one or more methods to be exposed as part of the stateful web service; See page 2, second clause of claim 1:

> identifying a member variable declared to implement said callback method

Also see third clause of claim 1: "conversational state"

in response to user input, automatically specifying, by the integrated development environment of the computing device, one or more declarative annotations, the declarative annotations, when recognized by a compiler, causing the compiler to generate one or more persistent components to maintain conversational state related to the identified method. See third clause of claim 1:

> specifying one or more declarative annotations associated with said callback method to cause a compiler to generate one or more persistent components to maintain conversational state related to the identified member variable.

The '807 application does not expressly disclose "identifying one of said one or more methods". However, it does teach "identifying a member variable." In this context, the "member variable" is implementing a method. As such, this "member variable" can be interpreted as a method. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the '807 application's "member variable" with the present application's methods. One of ordinary skill would have been motivated to provide support for callback methods as well as any other type of method.

This is a provisional obviousness-type double patenting rejection because the conflicting claims have not in fact been patented.

10. Claims 1-4, 10, 12, 15-17, 22, 31, 32, 34, 36, 38, 39, and 44 are provisionally rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claim 1-8, 19-23, 26, 27, 31-36, 38, 39, 43, and 44 of copending Application No. 10/784,492 (hereinafter "the '492 application"). Although the conflicting claims are not identical, they are not patentably distinct from each other because, for example, the '492 application discloses:

1. A method of specifying a stateful web service within a procedural programming environment (see page 39 lines 4-12), *the method comprising:*

first facilitating, by an integrated development environment of a computing device, a user in providing a source code representation of at least a portion of web service logic, the logic including one or more methods; See page 36 lines 2-4:

> an annotated source code, which is a programming language augmented with declarative meta-data capable of exposing program logic as a network-accessible service

Also see claim 5, e.g. "integrated development environment."

second facilitating, by the integrated development environment of the computing device, the user in identifying one of said one or more methods to be exposed as part of the stateful web service; See page 36 lines 5-6:

> at least one deployed service component capable of providing the network-accessible service to a client

Further, see page 37 lines 5-6:

> the annotated source code is capable of facilitating access to an external service, which can be one of stateful, stateless, synchronous, and asynchronous.

in response to user input, automatically specifying, by the integrated development environment of the computing device, one or more declarative annotations, the

declarative annotations, when recognized by a compiler, causing the compiler to generate one or more persistent components to maintain conversational state related to the identified method. See page 36 lines 7-10:

> an enhanced compiler capable of analyzing the annotated source code, recognizing numerous types of **meta-data annotations**, and generating a mechanism, which can include one or more of: object files, software components and deployment descriptors, to facilitate the deployment of the at least one service **component**

Further, see page 36 lines 14-16:

> the system is capable of simultaneously managing multiple transactions, wherein each transaction can be a **conversation** of a request and/or a response from the client for the network-accessible service.

This is a provisional obviousness-type double patenting rejection because the conflicting claims have not in fact been patented.

Claim Rejections - 35 USC § 102

11. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

12. Claims 1, 4, 10, 38, 39, and 44 are rejected under 35 U.S.C. 102(b) as being anticipated by prior art of record "Using WebLogic Enterprise JavaBeans" by BEA Systems (hereinafter "WebLogic").

In regard to claim 1, WebLogic discloses:

A method of specifying a stateful web service within a procedural programming environment, (See page 5 steps 1-3) *the method comprising:*

first facilitating, by an integrated development environment of a computing device, a user in providing a source code representation of at least a portion of web service logic, the logic including one or more methods; See section III on page 4:

> There are three parts to using WebLogic EJB:
> 1. Develop an EJBean or obtain one from a third-party supplier.

Also see page 3 paragraph 5 for disclosure of methods in an EJB:

> ...an EJBean contains the business logic (**methods**)...

Further, WebLogic discloses "a framework for the development and deployment of EJBeans" by a user/developer (see Applicants' comments at the bottom of page 12, filed 4/19/07), and is interpreted as providing an integrated development environment.

second facilitating, by the integrated development environment of the computing device, the user in identifying one of said one or more methods to be exposed as part of the stateful web service; See page 2:

> Session beans (either **stateful** or stateless) [emphasis added]

Also see page 3, 4th paragraph:

> With the EJB model, you can write or buy business components (such as invoices, bank accounts and shipping routes) and, during **deployment** into a certain project, specify how the component should be used -- which users have **access to which methods**, whether the framework should automatically start a transaction or whether it should inherit the caller's transaction, and so on. [emphasis added]

Also page 6 "Step 2" discloses identification of methods to be exposed:

> Check the deployment descriptor and modify any of its properties for your particular deployment (if required).

in response to user input, automatically specifying, by the integrated development environment of the computing device, one or more declarative annotations, the declarative annotations, when recognized by a compiler, causing the compiler to generate one or more persistent components to maintain conversational state related to the identified method. See page 5 "Step 2":

> The **Deployment Descriptor** ties together the different classes and interfaces, and **is used to build the code-generated class files**. It also allows you to specify some aspects of the EJBean's **deployment** at runtime.
>
> ...WebLogic EJB includes a utility application DDCreator that **takes a text file specification and creates the appropriate serialized deployment descriptor.**
> [emphasis added]

Also see page 6 along with "Step 3":

> Generate the wrapper classes using the WebLogic EJB **compiler** (ejbc)...
> This will **create the appropriate files** for the bean...
> [emphasis added]

Also, page 8 discloses the general capabilities of EJBeans with respect to persistence and transactional, or "conversational", state:

> An entity EJBean can save its state in any **transactional** or non-transactional **persistent** storage...
> [emphasis added]

In regard to claim 4, the above rejection of claim 1 is incorporated. WebLogic further discloses: *wherein the one or more declarative annotations are specified outside of the source code representation and associated with the identified method by the compiler.* See page 6 "Step 3".

In regard to claim 10, the above rejection of claim 1 is incorporated. WebLogic further discloses: *wherein the one or more declarative annotations are manually specified by a developer.* See page 6 "Step 2".

In regard to claim 38, WebLogic discloses:

An article of manufacture comprising: a storage medium having stored therein a plurality of programming instructions. Page 7 step 5 shows a directory path for storage of programming instructions as cited in the rejection of claim 31. All further limitations have been addressed in the above rejection of claim 16.

In regard to claim 39, the above rejection of claim 38 is incorporated. All further limitations have been addressed in the above rejection of claim 17.

In regard to claim 44, the above rejection of claim 38 is incorporated. All further limitations have been addressed in the above rejection of claim 22.

Claim Rejections - 35 USC § 103

13. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

14. Claims 2 and 3 are rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic as applied to claim 1 above, and further in view of "EJBDoclet", December 21 2000, by dreamBean Software (hereinafter "dreamBean").

In regard to claim 2, the above rejection of claim 1 is incorporated. WebLogic does not expressly disclose: *wherein the one or more declarative annotations are specified within the source code representation*. However, in an analogous environment, dreamBean teaches a declarative annotation within the source code. See page 2 under the heading "Custom EJBDoclet tags". It would have been obvious to one of ordinary skill in the art at the time the invention was made to use dreamBean's declarative annotations within WebLogic's source code. One of ordinary skill would have been motivated to automatically generate a remote interface (see dreamBean page 1 under "Features").

In regard to claim 3, the above rejection of claim 2 is incorporated. WebLogic does not expressly disclose declarative annotations within a comment field preceding an identified method. However, in an analogous environment, dreamBean teaches a tool for generating "EJB files from a commented bean source-file" (page 1 paragraph 1). dreamBean further teaches using comment fields preceding a method to support generation of externally accessible methods. See middle of page 5, where the annotation "@remote-method" appears in a comment preceding the identified methods "deposit" and "withdraw". It would have been obvious to one of ordinary skill in the art at the time the invention was made to use dreamBean's declarative annotation with WebLogic's

source code. One of ordinary skill would have been motivated to automatically generate a remote interface (see dreamBean page 1 under "Features").

15. Claims 5-8, 18, 40 are rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic as applied to claim 1 above, and further in view of "Enterprise JavaBeans" by Monson-Haefel (hereinafter "Monson-Haefel").

In regard to claim 5, the above rejection of claim 1 is incorporated. WebLogic further discloses use of the "ejbCreate" function. See page 24, 4th paragraph. WebLogic does not expressly disclose specifics of the start, continue or finish methods. However, in an analogous environment, Monson-Haefel teaches that "deployment descriptors" could be used as declarative annotations that indicate: *wherein the start method applies to the start of a stateful conversation between a client and the web service* (see section 7.4.2.1), *the continue method applies to the continuation of an ongoing stateful conversation between a client and the web service* (see section 7.4.2 "ejbActivate()" on page 4 of section 7.4), *and the finish method applies to the completion of an ongoing stateful conversation between a client and the web service* (see section 7.4.2.3). Monson-Haefel further teaches that such methods can be entered as annotations in section 10.6.3.2. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Monson-Haefel's teaching of stateful sessions with BAE Websphere's components. One of ordinary skill would have been motivated to provide a

dedicated stateful session bean to act on behalf of a client for its entire life cycle (see section 7.3 paragraph 1).

In regard to claim 6, the above rejection of claim 5 is incorporated. WebLogic does not expressly disclose: *wherein when a method declared to be a start method is invoked at run-time, a new instance of a conversation is created, and a unique identifier is associated with that conversational instance to facilitate management of multiple simultaneous conversations*. However, Monson-Haefel teaches instantiation of a conversation and return of an identifier upon invocation of a start method. See Section 7.4.2.1.

In regard to claim 7, the above rejection of claim 5 is incorporated. WebLogic does not expressly disclose: *wherein when a method declared to be a continue method or a finish method is invoked at run-time, a unique identifier provided by the client is obtained and used to access a corresponding instance of a conversation*. However, Monson-Haefel teaches the return of an identifier as noted in the above rejection of claim 6. Use of the representative identifier is inherent in referencing the session as discussed in section 7.4.2.3.

In regard to claim 8, the above rejection of claim 7 is incorporated. WebLogic does not expressly disclose: *wherein when a finish method is invoked at run-time, the corresponding instance of the conversation is destroyed after processing by the web*

service logic. However, Monson-Haefel teaches that the instance is destroyed after processing. See section 7.4.2.3.

In regard to claim 18, the above rejection of claim 16 is incorporated. All further limitations have been addressed in the above rejection of claim 5.

In regard to claim 40, the above rejection of claim 38 is incorporated. All further limitations have been addressed in the above rejection of claim 18.

16. Claim 9, 16, 17, and 41 are rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic as applied to claim 1 above, and further in view of prior art of record U.S. Patent 5,812,768 to Pagé et al. (hereinafter "Page").

In regard to claim 9, the above rejection of claim 1 is incorporated. WebLogic does not expressly disclose: *wherein the one or more declarative annotations indicate to the compiler whether the identified method is buffered, wherein if the identified method is buffered the compiler instantiates one or more queues to temporarily store one or more requests for the identified method*. However, in an analogous environment, Page teaches that interaction with web services can be implemented as buffered messages that operate asynchronously via message queues. See column 6 lines 39-46. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use

Page's queues with WebLogic's methods. One of ordinary skill would have been motivated to implement "store and forward" technology in order to provide reliable data delivery as suggested by Page (see column 2 lines 30-34).

In regard to claim 16, WebLogic discloses:

In a procedural programming environment, a method of generating a stateful web service (See pages 6 and 7), *the method comprising:*

reading on one or more computing devices a segment of procedural source code representing at least a portion of the web service; parsing on one or more computing devices the segment of source code to identify the presence of one or more declarative annotations identifying an associated method within the segment as being stateful; generating on one or more computing devices one or more object codes defining one or more publicly accessible service components based at least in part upon the source code; See page 6 "Step 3":

> **Generate the wrapper classes** using the WebLogic EJB compiler (ejbc) with this command (typed on one line), **referencing the serialized deployment descriptor**:
> $ java weblogic.ejbc -d /weblogic/myserver/temp AccountBeanDD.ser
> ...
> This will create the appropriate files for the bean, and place them in a temporary directory

Reading and parsing source code is an inherent feature of a compiler, as object code could not be generated without both steps. This passage also shows use of a computing device by the invocation of a command that is "typed on one line".

Generating on one or more computing devices meta-data based at least in part upon the one or more declarative annotations; associating on one or more computing devices meta-data with the one or more object codes; and See page 5 "Step 2":

> The Deployment Descriptor **ties together** the different classes and interfaces, and is **used to build** the code-generated class files.

All further limitations have been addressed in the above rejection of claim 9.

In regard to claim 17, the above rejection of claim 16 is incorporated. All further limitations have been addressed in the above rejection of claim 1.

In regard to claim 41, the above rejection of claim 38 is incorporated. All further limitations have been addressed in the above rejection of claim 19.

17. Claims 12, 31, and 34 rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic as applied to claim 1 above, and further in view of U.S. Patent 6,230,160 to Chan et al. (hereinafter "Chan").

In regard to claim 12, the above rejection of claim 1 is incorporated. WebLogic does not expressly disclose: *wherein said input includes graphical manipulation of the identified method by the developer via the integrated development environment.* However, in an analogous environment, Chan teaches an IDE for manipulation of program elements and methods. See FIG. 4A and column 8 lines 19-30.

In regard to claim 31, WebLogic discloses:

a storage medium; and a plurality of programming instructions stored on the storage medium (page 7 step 5 shows a directory path for storage of programming

instructions), *to provide an integrated development environment to facilitate a user in providing input, and* WebLogic discloses "a framework for the development and deployment of EJBeans" by a user/developer (see Applicants' comments at the bottom of page 12, filed 4/19/07), and is interpreted as providing an integrated development environment.

automatically specify, in response to the user input, one or more declarative annotations associated with an identified method of a stateful web service page 2:

> Session beans (either **stateful** or stateless) [emphasis added]

Also page 6 "Step 2" discloses identification of methods to be exposed:

> Check the deployment descriptor and modify any of its properties for your particular deployment (if required).

the declarative annotations, when executed by a compiler, causing the compiler to generate one or more persistent components to maintain conversational state related to the identified method. See page 5 "Step 2":

> The **Deployment Descriptor** ties together the different classes and interfaces, and **is used to build the code-generated class files**. It also allows you to specify some aspects of the EJBean's **deployment** at runtime.
>
> ...WebLogic EJB includes a utility application DDCreator that **takes a text file specification and creates the appropriate serialized deployment descriptor**.
> [emphasis added]

Also see page 6 along with "Step 3":

> Generate the wrapper classes using the WebLogic EJB **compiler** (ejbc)...
> This will **create the appropriate files** for the bean...
> [emphasis added]

In regard to claim 34, the above rejection of claim 31 is incorporated. All further limitations have been addressed in the above rejection of claim 4.

18. Claims 13, 20, and 42 are rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic as applied to claim 1 above, and further in view of the "Background of the Invention" section appearing on pages 1-3 of the originally filed specification (hereinafter "BOTI").

In regard to claim 13, the above rejection of claim 1 is incorporated. WebLogic does not expressly discloses: *a proxy object designed to facilitate interaction by the web service with one of an external web service or client.* However, BOTI teaches implementation of proxy objects. See page 2 lines 17-19. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use BOTI's teaching of implementation of a proxy object with WebLogic's web service. One of ordinary skill would have been motivated to automatically generate a required proxy mechanism.

In regard to claim 20, the above rejection of claim 16 is incorporated. All further limitations have been addressed in the above rejection of claim 13.

In regard to claim 42, the above rejection of claim 38 is incorporated. All further limitations have been addressed in the above rejection of claim 20.

19. Claim 14 is rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic and BOTI as applied to claim 13 above, and further in view of Page.

In regard to claim 14, the above rejection of claim 13 is incorporated. All further limitations have been addressed in the above rejection of claim 9.

20. Claims 15, 21, and 43 are rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic and BOTI as applied to claims 13, 20, and 42 above, and further in view of Monson-Haefel.

In regard to claims 15 and 21, the above rejection of claims 13 and 20 are respectively incorporated. All further limitations have been addressed in the above rejection of claim 6.

In regard to claim 43, the above rejection of claim 42 is incorporated. All further limitations have been addressed in the above rejection of claim 21.

21. Claims 32 and 33 rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic and Chan as applied to claim 31 above, and further in view of dreamBean.

In regard to claim 32, the above rejection of claim 31 is incorporated. All further limitations have been addressed in the above rejection of claim 2 and 12.

In regard to claim 33, the above rejection of claim 32 is incorporated. All further limitations have been addressed in the above rejection of claim 3.

22. Claim 35 is rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic and Chan as applied to claim 31 above, and further in view of BOTI.

In regard to claim 35, the above rejection of claim 31 is incorporated. All further limitations have been addressed in the above rejection of claim 13.

23. Claim 36 is rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic, Chan and BOTI as applied to claim 35 above, and further in view of Page.

In regard to claim 36, the above rejection of claim 35 is incorporated. All further limitations have been addressed in the above rejection of claim 14.

24. Claim 37 is rejected under 35 U.S.C. 103(a) as being unpatentable over WebLogic, Chan, and BOTI as applied to claim 36 above, and further in view of Monson-Haefel.

In regard to claim 37, the above rejection of claim 35 is incorporated. All further limitations have been addressed in the above rejection of claim 15.

Conclusion

25. Any inquiry concerning this communication or earlier communications from the examiner should be directed to J. Derek Rutten whose telephone number is (571)272-3703. The examiner can normally be reached on M-F 7:00-3:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571)272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

jdr

TUAN DAM
SUPERVISORY PATENT EXAMINER